CS485/685 Computer Vision
Spring 2012 – Dr. George Bebis
Programming Assignment 1
Due Date: 3/6/2012

In this assignment, you will experiment with image smoothing and edge detection. Please note that when displaying your results, all pixel values must be in the range [0, 255]. If pixel values fall outside this range, you must normalize them before displaying your results (e.g., use $I_{norm}(x,y) = 255(I(x,y) - \min)/(\max - \min)$).

1. Gaussian Smoothing

(a) Implement 1D Gaussian smoothing using convolution. Apply 1D Gaussian convolution on the 1D data in "Rect_128.dat" ("box" function; available from the course's webpage). Show and discuss your results using σ=1, 5, and 11. Use the code provided on my slides to generate the 1D Gaussian masks needed in your experiments (mask size = 5σ).

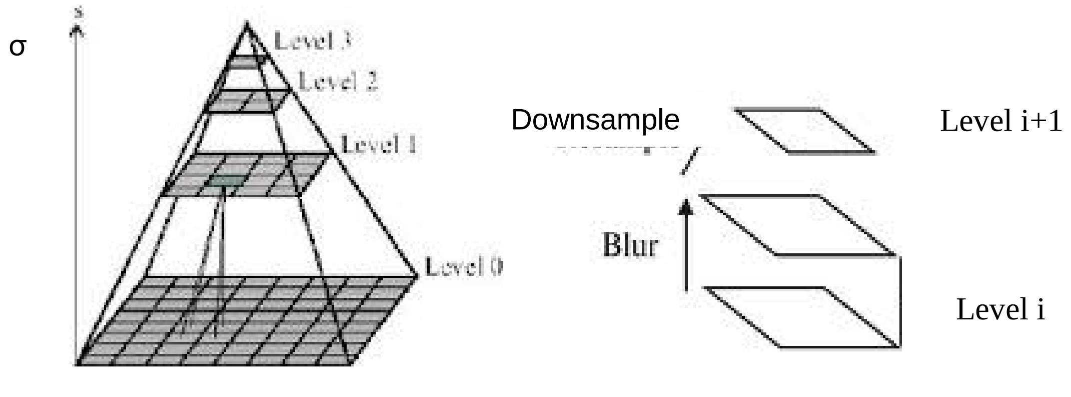
(b) We have discussed in class that $G_\sigma(x) * G_\sigma(x) = G_{\sqrt{2}\sigma}(x)$. Using σ=5, first compute $I_1(x) = (I(x) * G_\sigma(x)) * G_\sigma(x)$ where $I(x)$ corresponds to the data provided in "Rect_128.dat". Then, compute $I_2(x) = I(x) * G_{\sqrt{2}\sigma}(x)$. Show and discuss your results in each case; compare $I_1(x)$ with $I_2(x)$

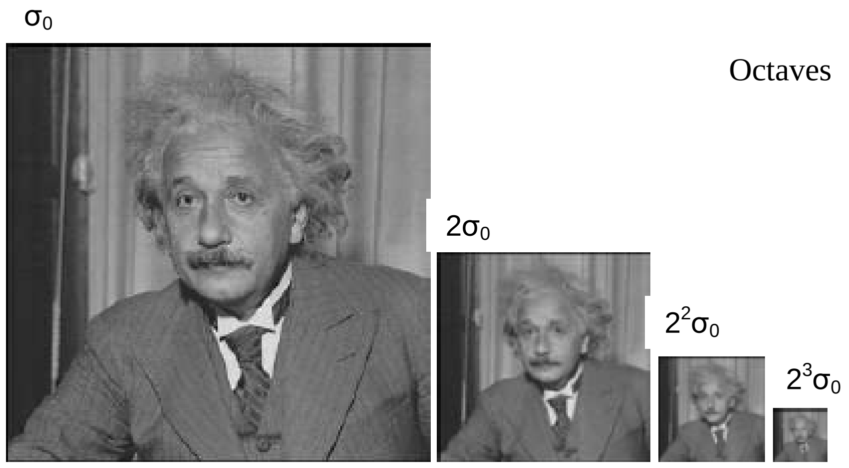
(c) Implement 2D Gaussian convolution. Extend the code provided on my slides to generate 2D Gaussian masks. Apply 2D Gaussian convolution using 2D masks on the *lenna* image using σ=1, 5, and 11. Show and discuss your results.

(d) Implement 2D Gaussian convolution using 1D Gaussian convolution as discussed in class. Apply 2D Gaussian convolution using 1D masks on the *lenna* image using σ=1, 5, and 11. Compare your results with those in (c).

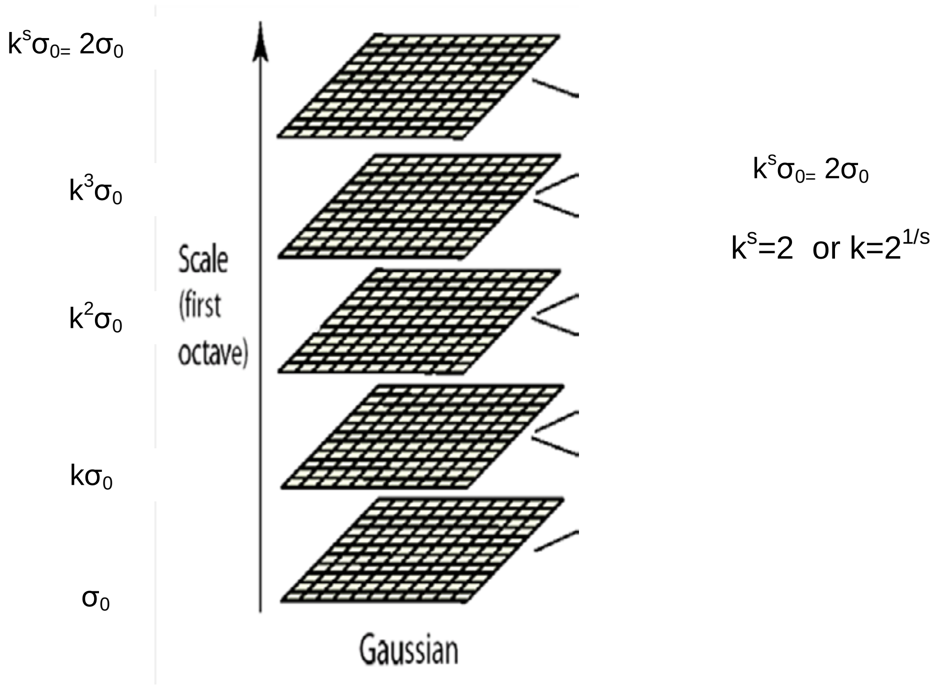
2. Gaussian Pyramid

In this part, you will write code to compute the Gaussian pyramid of an image. The Gaussian pyramid consists of smoothed, downsampled images of the preceding level of the pyramid, where the base level is defined as the original image (see figure below).



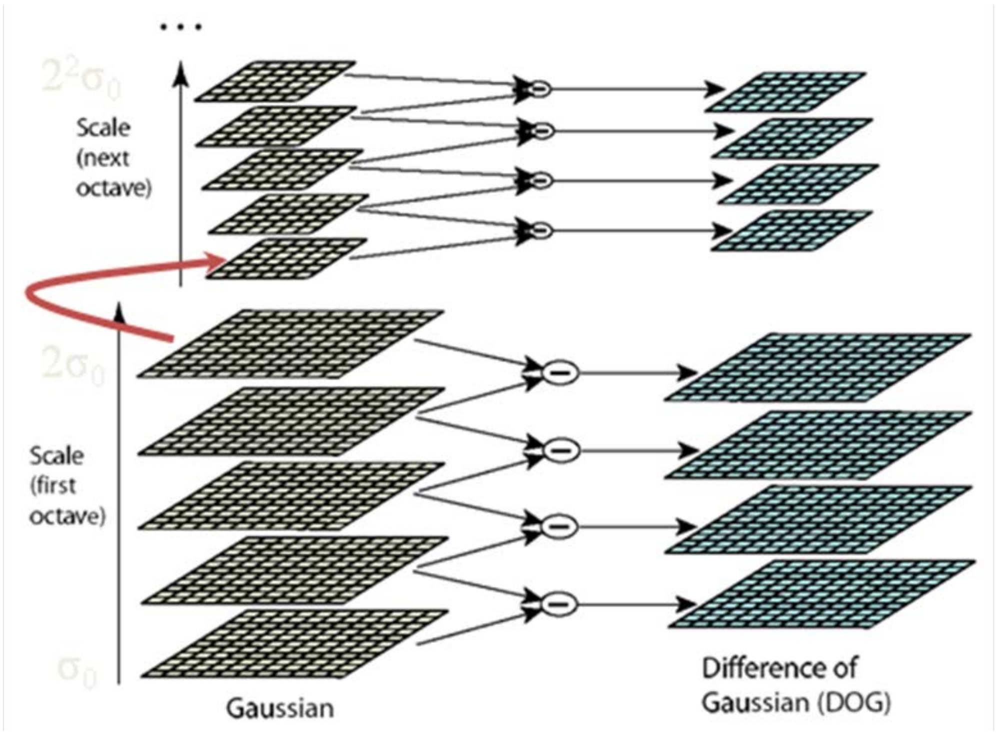


To create the Gaussian pyramid, start from some initial scale σ_0 and iterate n times by increasing the scale by a factor 2 each time (i.e., σ_0, $2\sigma_0$, $2^2\sigma_0$, …, $2^n\sigma_0$). The initial scale σ_0, and the number of levels in the pyramid should be parameters of your program. We refer to each level of the pyramid as "**octave**" (i.e., doubling of σ_0). Each octave should consist of s **intermediate** levels; the scale ratio k (i.e., step) between levels in the same octave should be $k=2^{1/s}$ as shown below (s should be a parameter in your program). Show your results on the *lenna* image.



3. Difference-of-Gaussians (DoG) Pyramid

In this part, you will write code to compute the DoG pyramid of an image. The DoG pyramid can be built by subtracting consecutive levels of the Gaussian pyramid (see figure below). As discussed in class, the difference between two Gaussian functions approximates the Laplacian of Gaussian (LoG). Therefore, the DoG pyramid approximates the LoG pyramid. Note that in order to build a DoG pyramid that uses all the levels of the Gaussian pyramid, you would need to create some extra levels in the Gaussian pyramid (i.e., one below the first level and one above the top level). Show your results on the *lenna* image.



4. Edge Detection

Implement edge detection using the Sobel masks for horizontal (S_y) and vertical (S_x) edges (use the masks provided on my slides). Given an image I(x,y), your program should compute and display the following images: **(a)** the gradient image in the x direction: $I_x(x,y)=I(x,y)*S_x(x,y)$, **(b)** the gradient image in the y direction: $I_y(x,y)=I(x,y)*S_y(x,y)$, **(c)** the gradient magnitude image: $M(x,y)=|I_x(x,y)|+|I_y(x,y)|$ and **(d)** the gradient direction image: $R(x,y)=a\tan 2(I_y(x,y)/I_x(x,y))$, and **(e)** edge image by thresholding M(x,y) (the threshold should be specified by the user). Show your results on the *lenna* and *sf* images.

4. Multiscale Edge Detection (Graduate Students Only).

We discussed in class that edges can occur at different scales. In this part, you will implement a multi-scale edge detection algorithm which consists of the following steps:

Step 1: *Given an image, generate its* DoG pyramid.

Step 2: *Threshold the DoG images* by assigning 1 to all pixels of magnitude greater than 0 and 0 to all pixels of magnitude less than or equal to zero.

Step 3: *Detect the "zero crossing" in the thresholded images.* This can be done by tagging any pixel which has at least one neighbor who is of different value than the pixel itself.

Step 4: *Examine the pixels surrounding the zero crossing pixels* in the DoG image. Calculate the local variance by considering a neighborhood of 5x5 pixels. To calculate the local variance, first find the mean of the DoG image in the local neighborhood. Then find the sum of the square of the difference of the value of the pixel from this mean in this neighborhood. If the value is greater than a certain threshold (user specified), detect this pixel as an edge pixel.

Show your results on the *lenna* and *sf* images.

Laboratory Write-up

For each programming assignment, you are to turn in a brief report (see instructions posted on the course's website). The report is very important in determining your grade for the programming assignment. Be well organized, type your reports, and include figure captions with a brief description for all the figures included in your report. Motivation and initiative are greatly encouraged and will earn extra points.